BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images may include (but are not limited to):

- BLACK BORDERS
- TEXT CUT OFF AT TOP, BOTTOM OR SIDES
- FADED TEXT
- ILLEGIBLE TEXT
- SKEWED/SLANTED IMAGES
- COLORED PHOTOS
- BLACK OR VERY BLACK AND WHITE DARK PHOTOS
- GRAY SCALE DOCUMENTS

IMAGES ARE BEST AVAILABLE COPY.

As rescanning documents *will not* correct images, please do not report the images to the Image Problem Mailbox.

社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE.
SSE98-72, IN98-53(2), CS98-69(1998-09)

IP経路制御におけるアドレス解決エンジン

山田　憲晋　　西原　基夫　　升田　道雄　　林　偉夫　　村上　紅

NEC　C&C基盤開発研究所　第一研究部
〒270-1198　千葉県我孫子市日の出1131
phone: 0471-85-6736
e-mail : kenshin@ptl.abk.nec.co.jp

昨今のインターネットの隆盛にともない、バックボーンネットワークに配備されるIPパケット転送装置は、OC12/OC48といった高速な伝送インタフェース上において極めて高速な経路検索を行う能力が必要とされ本論文は上記を実現するため、従来ソフトウエアで行われていた二分木処理アルゴリズムの改良を行い、ハードウエア回路と通常のメモリのみで簡易に実現する手段を提案する。さらに試作の結果として、PLD＋メモリの構成で、128kエントリの経路に対し完全にCIDRに準拠しつつ数百Mbpsの伝送速度に対応可能であることを報告する。

キーワード：インターネット、ルーティング、アドレス解決、CIDR、二分木

Address Resolution Engine for IP Routing

Kenshin Yamada　Motoo Nishihara　Michio Masuda　Takeo Hayashi　Kurenai Murakami

C&C Network Products Development Laboratories, NEC　Corporation
1131, Hinode, Abiko, Chiba, 270-1198, Japan
phone:+81 471 85 6736
e-mail:kenshin@ptl.abk.nec.co.jp

As the Internet traffic increases exponentially, network equipment to transmit IP packets on the backbone network need to support OC12/OC48 interface and resolute IP routes at wire speed, which is almost impossible for conventional routers executing their resolution by software algorithm. This paper proposes the improved binary tree algorithm, which is more suitable for hardware platform, and an address resolution engine with the simple ASIC and standard memories to utilize its algorithm. Our address resolution engine has 128k route entries, fully conforms to CIDR specification and can perform wire-speed searching even on a few hundred Mbps link rate.

Keyword : Internet, Routing, Address Resolution, CIDR, Binary Tree

1 まえがき

IP経路の検索は、宛先IPアドレスの属するネットワークアドレスを解決することにより実現する。過去のクラス構造に基づくネットワークではIPアドレスの上位数ビットを参照することにより容易にネットワークアドレスを識別することが可能であった。しかし、IPアドレスの枯渇を解決するために導入された CIDR(Classless Inter-Domain Routing)[1]の普及によりクラス構造は消滅し、ネットワークアドレスを容易に判別することが出来なくなった。CIDRに対応したネットワークでは、図1に示すように検索IPアドレスが複数の経路エントリと一致する複数マッチングが発生しうる。複数マッチングが発生した場合、もっともマスク長の長い経路を選択しなければならない(最長ネットマスク選択の原則)。

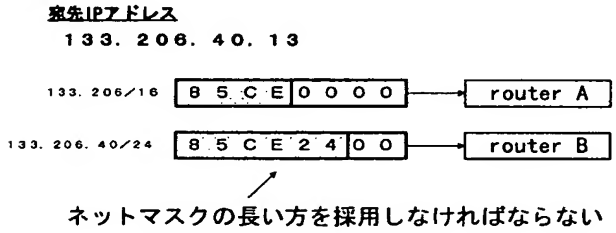

図 1 複数マッチングの例

CIDRの普及により、従来ソフトウエアで行われていたハッシュテーブルを用いた経路検索技術は有効に機能しなくなった。現在では、CIDRに対応した経路を効率よく検索するために二分木を用いた検索機構がFreeBSD等に実装されている[2][3]。

しかし、現状の指数関数的なネットワークトラフィックの増大により、もはやコア網を流れるトラフィックをソフトウエアで処理することは限界に近づきつつある。本論文では、ソフトウエアで利用される二分木検索アルゴリズムの改良を行い、ハードウエアで高速に処理する手法の検討を行い、アドレス解決エンジンを試作したので報告する。

2 二分木による検索アルゴリズム

本章では、二分木を用いた検索アルゴリズムに関して説明する。

2.1 二分木の構成

経路テーブルから二分木を構成する場合、各経路エントリの宛先ネットワークアドレスを単なるビット列と見なして、図 2に示すような二分木を作成する。本二分木は、必要のない分岐点を削減し、エントリ数を最適化している。

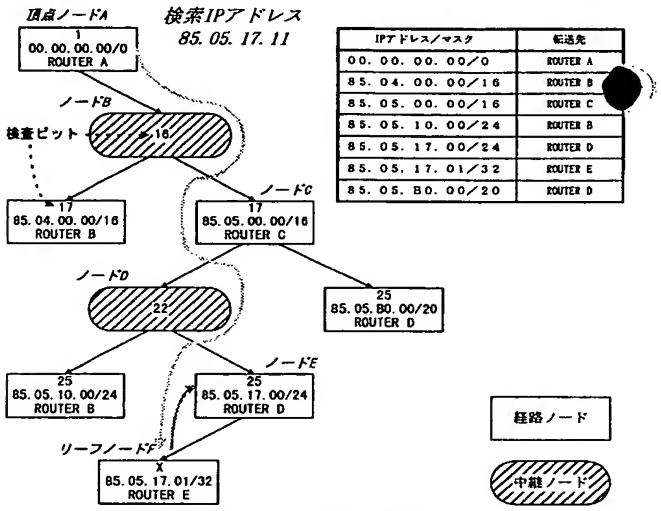

図 2 二分木の例

ツリーを構成する各ノードは最大 2 つのノード(以下、子ノードと呼ぶ)へ分岐する。ノードには、経路情報を有する経路ノードと、経路情報を持たず分岐専用に使用される中継ノードの2種類がある。中継ノード数が経路ノード数を越えることは無いため、ツリーを構成するノード数は経路エントリ数の 2 倍あれば十分である。複数マッチングの発生する経路において、マスクの短い方の経路は、頂点ノードとリーフノードの間の分岐点に位置するが、他の経路ノードは全てリーフノードである。 また、各ノードは検査ビット番号を有している。検査ビット番号は、各ノードにおいてどちらの子ノードに分岐すべきかを判定するビット位置を示すものである。

2.2 二分木の検索手法

二分木を用いた経路検索方法は次のようになる。図 2 の経路ツリーにおいて検索 IP アドレス 85.05.17.11（便宜上 16 進数表記をしている）が入力された場合を考える。各ノードは、検索 IP が入力されると検査ビット番号で指定される検索 IP アドレスの 1 ビットを評価し、左右どちらに分岐すべきかを決定する。例えば、ノード B の検査ビット番号は 16 であるので、検索 IP アドレス 85.05.17.11 の 16 ビット目が 1 であることから右側の子ノードへ進む。本処理を繰り返すことにより、検索 IP アドレス 85.05.17.11 は、頂点ノード A からリーフノード F へ到達する。

検索 IP アドレス 85.05.17.11 がリーフノード F へ到達する間に通過する経路ノードは、図 3に示す 4 ノードである。この全てのノードとアドレス比較を行った場合、検索 IP アドレス 85.05.17.11 は、頂点ノード A、ノード C、ノード E とは一致するが、リーフノード F とは一致しない。よって、最長ネットマスク選択の原則より、一致するノードの中でもっともマスク長の長いノード E の経路情報を選択しなければならない。

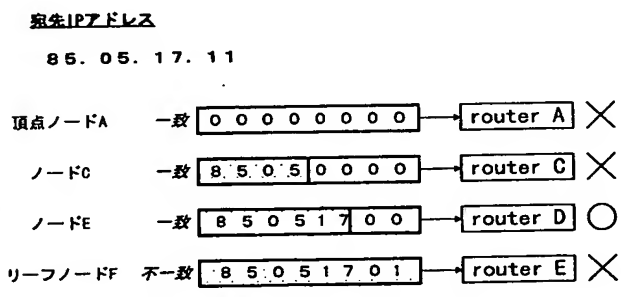

図3　通過する経路ノード

ソフトウエアで検索を行う場合、アドレス比較の回数を減少させるために、バックトラックという手法を用いて経路検索を行う。検索はツリーを下方向に下っていく Forward Search と、通過したノードを逆方向に辿っていく Backward Search の 2 つのフェーズに分けられる。

Forward Search

検索 IP アドレスが入力されると、検査ビット番号を用いて子ノードの選択処理を繰り返しながらリーフノードへ到達する。頂点ノードからリーフノードへ到達する際に経路ノードを通過してもアドレス比較は行わない。

Backward Search

検索 IP アドレスとリーフノードの持つネットワークアドレスを比較する。一致する場合はノードの持つ経路情報を採用する。一致しない場合は、上方向のリンクに存在するノード（以下、親ノードと呼ぶ）を読み出す。

新たに読み出されたノードが経路ノードである場合、同様にアドレス比較を行い、一致する場合はノードの持つ経路情報を採用する。一致しない場合もしくは中継ノードである場合は、さらに親ノードを読み出す。以上の処理を繰り返すことにより、ツリーを逆方向に辿りながら検索 IP アドレスに一致する経路ノードを検索する。このようにツリーを逆方向に探索する操作をバックトラックと呼ぶ。

Forward Search のフェーズにおいては、経路ノードであるかどうかを判別する処理やアドレス比較処理は不要で、分岐処理のみを行えばよく、高速にリーフノードへ到達することが可能である。

Backward Search のフェーズにおいては、マスクの長い経路から順番にアドレス比較を行うため、アドレス比較が最初に一致したノードの経路情報を採用すれば良い。図 2の経路ツリーを検索 IP アドレス 85.05.17.11 で検索する場合、まずリーフノード F でアドレス比較が行われる。リーフノード F とは一致しないため、次にノード F の親ノードであるノード E とアドレス比較を行う。検索 IP アドレスはノード E と一致するため検索は終了し、ノード E の経路情報を採用する。よって、バックトラックを利用する場合、アドレス比較はノード F とノード E のみで行われる。

実際のコア網のルーティングテーブルで経路ツ

リーを構成した場合、経路エントリのほとんどはリーフノードである。よって、ほとんどの検索においてリーフノードで経路が解決するためバックトラックは発生しない。

このように、ソフトウエア処理ではバックトラックによる検索を利用して、アドレス比較処理の回数を削減し、効率的な経路検索を実現している。しかし、バックトラックによる検索では、検索IPアドレスに該当する経路が存在しない場合、ツリーの頂点までバックトラックにより戻っていくことになる。よって、最悪の場合ツリーを往復することになる。

3　二分木処理のハードウエア化

インターネットトラフィックの指数的な増大により、ソフトウエアにより処理ではもはやコア網を流れる膨大なトラフィックを処理することは困難になってきている。本章では、二分木検索処理をハードウエア化することにより検索時間を短縮化することを考える。

3.1 アルゴリズムの改良

3.1.1バックトラック操作の削減

経路検索のトータルの検索時間は、
「通過ノード数×1ノード処理時間」で算出される。

ソフトウエアでバックトラックにより検索を行っているのは、アドレス比較回数を削減し、1ノードあたりの処理時間を減少させるためである。

ノード処理のハードウエア化を行った場合、次ノード選択処理とアドレス比較処理は並列に処理可能である。よって、ノード処理時間はアドレス比較処理を実行するかどうかにかかわらず一定にできる。

前章で述べたように通過ノード数に関しては、バックトラックによる検索では最悪の場合ツリーを往復することになる。よって、Forward Searchと Backward Searchに分かれていた検索処理を全てForward Search内で実行することにより、通過ノード数の最大値はIPアドレスのビット数より32＋1＝33となる。

3.1.2中継ノードでのアドレス比較処理

アドレス比較処理が必要なのは経路ノードの持つ経路情報を採用すべきかどうかを判断するためであるので、中継ノードにおいてはアドレス比較処理を行う必要はない。しかしハードウエア回路では、中継ノードにおいても対応するネットワークアドレスを付加して経路ノードと同様にアドレス比較を行うことにより、検索の高速化が可能である。

図4は、図2に示した経路ツリーに対して、中継ノードにネットワークアドレス情報を付加したものである。検索ビット番号はネットワークアドレスのマスク値に1加えた値となる。

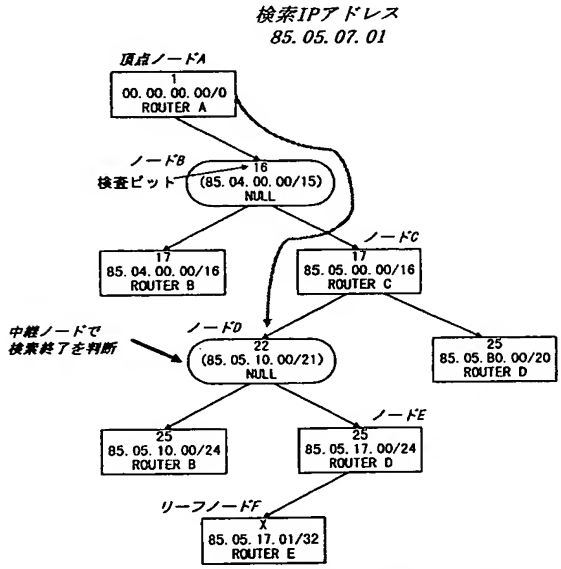

図4　ハードウエアによる検索例

Forward Searchのみで経路検索を行う場合、各ノードにおいてアドレス比較を実行するため、アドレス比較が不一致となった時点で、検索を終了することが出来る。中継ノードに対してもネットワークアドレスを設定しているので、中継ノードでアドレス不一致となった場合にも経路検索を終了することが可能である。図4の経路ツリーに検索IPアドレス85.05.17.01が入力された場合、中継ノードDにおいてアドレス比較が不一致となるため、検索終了する。ノードE及びリーフノードFを評価する必要はない。

採用する経路は、検索を終了した際に一番最後に

アドレス比較が一致したノードの経路情報である。よって、検索 IP アドレス 85.05.17.01 の場合、ノード B の経路情報を採用する。

3.2 回路構成・動作

以上に述べた処理を実現するためのハードウエアは、単純な検索回路とノード情報格納用のメモリを用いて実現可能である。検索処理回路のブロック構成図を図 5に示す。本回路は、次ノード選択回路、アドレス比較回路、検索終了判別回路の 3 回路から構成される。

1. 次ノード選択回路
   検索 IP アドレスの一ビットをマスク情報により選択し、選択された信号をセレクタ入力として、左子ノードもしくは右子ノードを選択する。
2. アドレス比較回路
   検索 IP アドレスと各ノードのネットワークアドレスを比較する。まず、検索 IP アドレスをマスク情報から算出されるマスクアドレスでマスクした後、アドレス部分の比較を行う。アドレス比較が一致し、かつ経路情報が存在する場合、経路情報番号をレジスタに保持する。
3. 検索終了判別回路
   検索を終了するかどうかの判別を行う。検索終了となる条件は、次ノード選択回路において次に読み出すべきノードが存在しなくなるか、アドレス比較回路においてアドレス比較が不一致となった場合のどちらかである。検索終了となった際に、保持されている経路情報を採用すべき経路情報として出力する。一致する経路が存在しなかった場合は、頂点ノードに設定された経路情報が出力される。

以上のような単純な回路を用いて二分木検索を実現することが可能である。

図 6 に、図 4 の経路ツリーを検索 IP アドレス 85.05.07.01 で検索した場合のタイムチャートを示す。

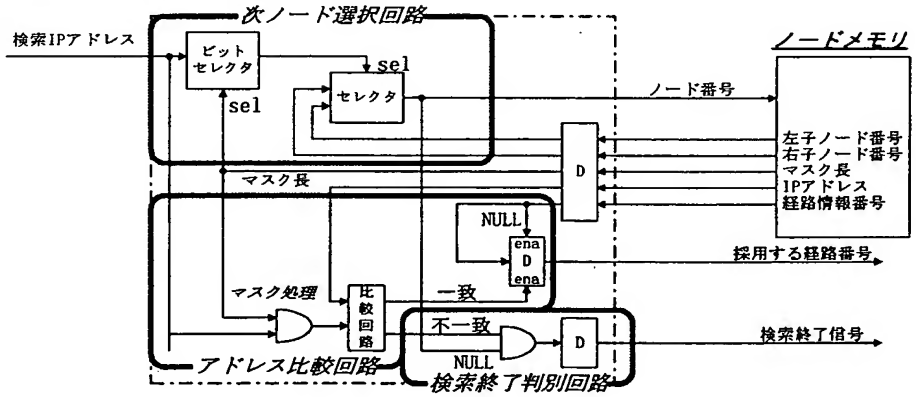

図5 検索回路のブロック構成

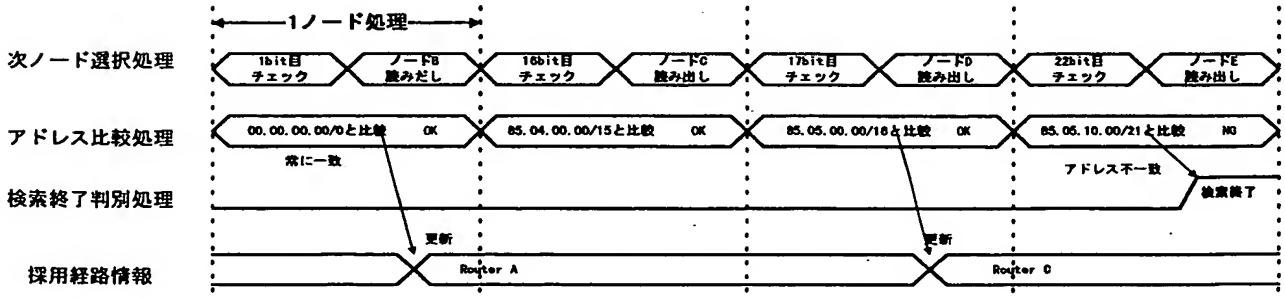

図6 経路検索のタイムチャート

4 アドレス解決エンジンの開発

以上の検討を踏まえアドレス検索エンジンの試作を行った。

本アドレス解決エンジンの基本性能を表 1 に示す。

表 1 アドレス解決エンジンの主要諸元

Item	Specification
Device	Programming Logic Device 1 約 35kGate
Memory	Static Ram 256k×32bit×3 Access Time = 20nsec
System Clock	19.44Mhz (51.44 nsec)
Routing Entries	128,000 entries

4.1 パイプライン処理による高速化

図 6 に示したタイムチャートにおいて、検索処理のクリティカルパスは「次ノード選択処理＋ノードメモリ読み出し」の部分にある。本アドレス解決エンジンでは、各ノードの有する子ノード情報として読み出しアドレスと共にマスク情報を格納しておき、1 サイクル早くマスク情報を読み出す。マスク情報をあらかじめ取得することにより、次ノード選択処理回路の処理は 2：1 セレクタにより子ノードを選択する回路のみになるため、クリティカルパスのほとんどはメモリ読み出し速度ということになる。

4.2 追加削除処理の実装

本アドレス解決エンジンは、経路エントリを追加･削除するための回路を有する。追加･削除処理を実装することにより、経路テーブルに変化が生じた際のノードメモリの更新処理を CPU から制御する必要が無く、高速に経路ツリーの更新を実現する。

4.3 経路情報の削除

本アドレス解決エンジンは CPU から検索 IP アドレスが入力されると、対応する経路情報へのポインタを解決して出力する。実際の経路情報は,CPU 内のメモリで管理されている。各ノードの読み出しアドレスと経路情報へのポインタを一致させることにより、ノードメモリに経路情報ポインタを格納する必要が無くなる。CPU 側ではノードの読み出しアドレスにあわせて、経路情報のテーブルを作成する必要がある。

4.4 処理性能

本アドレス解決エンジンは、1 ノード処理を 1 クロックサイクル 51. 44nsec で実行する。よって、最大検索時間は、51.44nsec×33=1.7usec である。平均検索時間に関しては、コア網におけるルーティングテーブル[4]より経路ツリーを構成し、ランダムに発生させた IP アドレスを用いて評価した結果、51.44nsec×13.23=0.68usec という結果を得ることが出来た。これは 1.5Mpps であり、OC12 インタフェースにおいて 64byte の IP パケットをワイヤー速度で処理可能な性能である。

5 まとめ

IP アドレス解決を高速に行うために、現在、二分木を用いた複雑な検索アルゴリズムがソフトウエアで実装されている。本論文では、二分木検索をのアルゴリズムの改良を行い、単純なハード回路と SRAM を用いてアドレス検索エンジンを試作した。本検索エンジンは、平均検索時間 0.68 usec でアドレス検索を実行する。これは、OC12 インタフェースにおいて、64byte の IP パケットをワイヤー速度で転送可能な処理能力である。今回の試作では低速な FPGA を用いて動作検証を行ったが、高速メモリを使用し、検索回路を LSI 化することによりさらなる高速化が可能である。

参考文献

[1] V. Fuller, T. Li, J. Yu and K. Varadhan, "*Classless Inter-Domain Routing(CIDR): an Address Assignment and Aggregation Strategy*", RFC1519, 1993.

[2] Keith Sklower." *A Tree-Based Routing Table for Berkeley Unix*", Technical report, University of California, Berkeley

[3] Kazuhiko Yamamoto, Akira Kato and Akira Watanabe, *"Radish - A Simple Table Structure for CIDR"*, Techinical Memorandum with the RADISH Lite 0.9 package, July 1995

[4] Merit. Routing table snapshot on 27 April 1998 at the Mae-East NAP. ftp://ftp.merit.edu/statisctics/ipma.

電子情報通信学会技術研究報告　信学技報 Vol. 98 No. 297
1998年9月24日 発行

IEICE Technical Report

発行人　東京都港区芝公園3丁目5番8号　機械振興会館内
社団法人 電子情報通信学会　事務局長　飯野　理

発行所　東京都港区芝公園3丁目5番8号
社団法人 電子情報通信学会　電話〔03〕3433－6691
郵便振替口座 00120-0-35300

The Institute of Electronics, Information and Communication Engineers, Kikai-Shinko-Kaikan Bldg., 5-8, Shibakoen 3 chome, Minato-ku, TOKYO, 105-0011 JAPAN